Programación

Cadenas

Manuel Molino Milla Luis Molina Garzón

26 de enero de 2015

Ejercicio 1

Crea una clase denominada *PalabraLeida*, que tenga como único atributo un *String* denominado *valor* y que contenga los siguientes métodos:

- NumeroDeLetras()
- EmpiezaPorVocal()
- AcabaEnVocal()
- NumeroDeVocales()
- ContieneH()
- EsUnPalindromo()
- SonIguales(String palabra):boolean (sin tener en cuenta mayúsculas o minúsculas)

Una palabra es un palíndromo si al leerla de izquierda a derechas es la misma palabra leida en sentido contrario, ejemplo *reconocer, rotor, salas, seres, somos.*
Posteriormente crea una clase denominada *TestPalabraLeida* que compruebe el funcionamiento de dicha clase y lea la palabra mediante la clase Scanner.
Para comprobar el funcionamiento del método *SonIguales(String palabra):boolean* utiliza los argumentos del programa principal para obtener el parámetro *String* de dicho método.
Genera la documentación de la clase *PalabraLeida* y el diagrama UML de la misma.
Crea un fichero jar ejecutable que compruebe el funcionamiento del *TestPalabraLeida*

Ejercicio 2

Crea una nueva clase denominada *ClaveSegura* que tenga como único atributo un *String* denominado clave. Puedes usar un *constructor* o un *setter* para la inicialización de dicho atributo.
Dicha clase contará con un método que se llame *esClaveSegura*. Una clave es segura si cumple los siguientes requisitos:

- Tenga al menos 8 caracteres.
- Tenga al menos una letra en minúscula.
- Tenga al menos una letra en mayúscula.
- Contenga al menos un número.
- Tenga al menos un carater no alfanumérico.

Posteriormente crea una clase *TestClaveSegura* que genere de forma aleatoria clave de longitud aleatoria (entre 0 y fuenteCaracteres.length()-1), muestre por pantalla dicha clave e indique si es segura o no.
Para obtener los caracteres que forman la clave se usará los caracteres del siguiente String:

> String final FUENTE_CARACTERES = "aAbBcCdDeEfFgGhHiIjJkKlLmMnNñÑoOpPqQrRsStTuUvVwWxXyYzZ0123456789¿?()=@.:,;!¡&{}";

Genera la documentación de la clase *ClaveSegura* y el diagrama UML de la misma.
Crea un fichero jar ejecutable que compruebe el funcionamiento del *TestClaveSegura*

Ejercicio 3

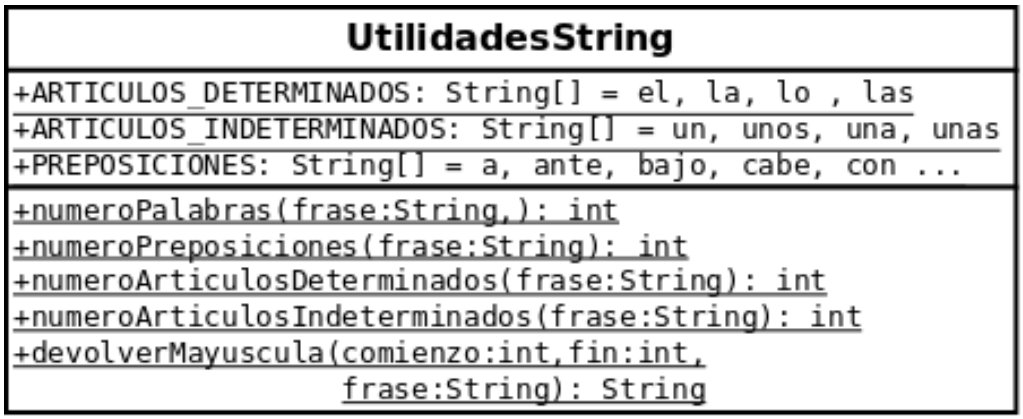
Queremos leer los datos de un fichero de texto pero sin tener que usar las clase que aporta java en relación a entrada y salida, para esto realizaremos el siguiente comando:

```
cat nombres_mujer.txt | java Programa
```

En el caso de windows cambia el comando cat por type.
El programa no va a usar el paradigma de POO y deberá hacer lo siguiente:

- Usaremos la clase Scanner para realizar la lectura.
- Indicar cuantas palabras ha leido.
- Crear dos listas una con aquellos nombres que empiezan por A y otra para aquellas palabras que no acaben en vocal. Posteriormente las mostramos por pantalla.

- Crea otra dos listas para guardar las palabras con mas y con menos letras. Ambas listas contendran String con el mismo tamaño. Posteriormente mostramos por pantalla ambas listas.
- En el caso que pasemos un argumento al programa, el programa no debe realizar nada de lo anterior y lo que debe hacer es compruebar si dicho argumento es un nombre que aparece en el fichero y nos diga por tanto que existe, o bien que nos sugiera nombre que empiezan por la dos primeras letras que el parámetro pasado.



Ejercicio 4

Igual que en el ejercicio anterior, lee el fichero *contitucion.txt*. Guarda cada palabra en un *ArrayList*. Posteriormente crea un *StringBuilder* en el que vas a añadir quinientos *String* del *ArrayList* inicial, la elección de la posición del *String* se hace de forma aleatoria entre los número 0 y el tamaño del *ArrayList* (puede haber repeticiones de palabras o posiciones). Todo esto lo realizas en una clase denominada *TestConstitucion.java*.
Posteriormente implementa la clase UtilidadesStrig.java de acuerdo a su *diagrama UML*.
En la clase *TestConstitucion.java* comprueba el funcionamiento de la clase anterior.
Para contar el número de articulos o preposiciones, puedes convertir el *String* en un array de *String* mediante el método *splice* de la clase *String*.
Genera la documentación de la clase *UtilidadesString.java*